Document made available under the Patent Cooperation Treaty (PCT)

International application number: PCT/SE05/000348

International filing date: 10 March 2005 (10.03.2005)

Document type: Certified copy of priority document

Document details: Country/Office: SE
Number: 0400622-7
Filing date: 12 March 2004 (12.03.2004)

Date of receipt at the International Bureau: 04 April 2005 (04.04.2005)

Remark: Priority document submitted or transmitted to the International Bureau in compliance with Rule 17.1(a) or (b)

World Intellectual Property Organization (WIPO) - Geneva, Switzerland
Organisation Mondiale de la Propriété Intellectuelle (OMPI) - Genève, Suisse

PCT / SE 2005 / 0 0 0 3 4 8

PRV

PATENT- OCH REGISTRERINGSVERKET
Patentavdelningen

Intyg
Certificate

Härmed intygas att bifogade kopior överensstämmer med de handlingar som ursprungligen ingivits till Patent- och registreringsverket i nedannämnda ansökan.

This is to certify that the annexed is a true copy of the documents as originally filed with the Patent- and Registration Office in connection with the following patent application.

PATENT- OCH REGISTRERINGSVERKET

(71)	Sökande Applicant (s)	Telefonaktiebolaget L M Ericsson (publ), Stockholm SE
(21)	Patentansökningsnummer Patent application number	0400622-7
(86)	Ingivningsdatum Date of filing	2004-03-12

Stockholm, 2005-03-15

För Patent- och registreringsverket
For the Patent- and Registration Office

Gunilla Larsson

Avgift
Fee

Prepared (also subject responsible if other)		Document No.		
Christer Boberg				
Approved	Checked	Date	Rev	P-Number
		2003-02-19	PA1	

INVENTION DISCLOSURE

Inventor Name	Corp. ID	Company	Nationality	Share	Signature
Christer Boberg	ERABOCH	EAB	SWE		
Anders Lindgren	ERAARLE	EAB	SWE		
Stefan Andersson	ETXSSAA	EAB	SWE		
Hans Lindgren	ERAHLIN	EAB	SWE		

Please underline a contact person for Patent Support. If no one is indicated we will contact the first inventor.

1 Title of the Invention

Pull / Push mechanism for IMS/SIP based list subscriptions

2 Background

2.1 Technical Background/Existing Technology

By subscribing to lists (pre-defined or ad-hoc) of users instead of subscribing to each user individually performance can be improved dramatically. This is already defined by IETF drafts http://www.ietf.org/internet-drafts/draft-ietf-simple-event-list-04.txt and http://www.ietf.org/internet-drafts/draft-camarillo-sipping-adhoc-management-00.txt.

The basic mechanisms for subscribing to information is defined by IETF RFC3265 and it is possible to achieve both a push behaviour, where information is pushed to the subscriber in real time, as well as a pull behaviour where the clients fetch information when needed.

Prepared (also subject responsible if other)		Document No.		
Christer Boberg				
Approved	Checked	Date	Rev	P-Number
		2003-02-19	PA1	

2.2 Problems with existing solutions

The existing solutions are useful in many cases but in some cases there might be other more optimal solutions.

One existing example is a pure push solution where a client subscribes to presence information for a list of users. The client will then receive notifications from the Resource List Server (RLS) when the state is changed for any of the user in the list. This might give a large amount of notifications between the RLS and the client and in a network, such as wire-less, where bandwidth is an issue this is a problem. Also battery capacity is a big issue since asynchronous notifications to a terminal will take power.

There are solutions defined to have a rate limitation functionality on the server side which is used to have a minimum time between two notifications to the same client. In a case where the client really needs real-time data, the rate limit value must be low and then the actual saving might be very low.

In these cases it is possible to poll for information when needed, or it is possible to use a combination of both push with rate limitation and poll when needed. However the poll solution have major drawbacks when subscribing to a list. In this case the RLS have to fetch information for all users in the list in real-time which both will extend the time for the actual poll and add a very high number of requests to fetch the data for each user. This is especially important to avoid in a multi-operator scenario where standard procedures must be followed and it is not possible for internal optimizations.

3 Basic Concept

This solution combines a pull mechanism between the client and the RLS and a push mechanism between the RLS and the resources hosting the users in the list.

Prepared (also subject responsible if other)		Document No.		
Christer Boberg				
Approved	Checked	Date	Rev	P-Number
		2003-02-19	PA1	

4 Detailed description

4.1 Detailed Technical Description of the Invention

The client indicates that he wants to subscribe for certain information for a list of users. The subscription is sent to the RLS including an expire time in the message which informs the RLS that it shall set up a dialogue for this subscription not only between the client and the RLS but also between the RLS and the servers hosting the users in the list. This time should be fairly long e.g. 12 hours. At the same time the client indicates that he don't want to have notifications sent to him more often than a specific throttle value is indicating as defined by IETF draft http://www.ietf.org/internet-drafts/draft-niemi-sipping-event-throttle-00.txt. This throttle time should also be fairly long and the recommendation is to use the same value as the subscription expire time.

By doing so the RLS will set up subscriptions with each resource using the subscription expire time but not including the throttle value since this is only valid for the notifications that shall be sent to the client. The RLS will then receive and cache state updates from the different resources but it will not send these out to the client until the throttle time elapse or a new subscribe is received. The subsequent subscribes shall have the same values set as the initial subscribe.

By doing so the client can poll for data when desired but the RLS will not have to go to each resource to fetch information and the actual fetch will be much faster and the amount of NNI traffic is reduced. The effect of this will be higher the more "external" users that are included in the list.

See also the slides for: Presence Optimizations for Push /Pull. 1

Another aspect, or optimization, is to sub-divide the list into "smaller sub-lists", comprising fewer amounts of users. These sub-lists could be more adapted to the type of service or type of application that needs the presence data. This sub-division of lists would reduce the number of notification between the RLS and the terminal even more, because presence data of all users on the list do not have to be sent, only presence data of users needed for that specific type of service/application.

4.2 Advantages of the Invention

Compared to a pure push solution it saves bandwidth over the air interface, it saves battery in the terminal and it prevents information from being pushed to a terminal when the information is not requested.

Prepared (also subject responsible if other)		Document No.		
Christer Boberg				
Approved	Checked	Date	Rev	P-Number
		2003-02-19	PA1	

Compared to a pure pull solution it saves bandwidth over the network to network interface (NNI) and it also saves time for doing the poll since the RLS already have updated data about each user and do not have to fetch it in real time.

The present invention relates to a method and an information delivery server for providing information to a client, as defined in the attached claims 1-16 and illustrated in Figure 1. 2

5 Core of the Invention

The existing standards and solutions (as far as we know) based on the IETF-SIP/SIMPLE and 3GPP does either use pure push or pure pull.

6 Abbreviations

NNI *Network to Network Interface*

7 References

http://www.ietf.org/internet-drafts/draft-ietf-simple-event-list-04.txt

http://www.ietf.org/internet-drafts/draft-camarillo-sipping-adhoc-management-00.txt

http://www.ietf.org/internet-drafts/draft-niemi-sipping-event-throttle-00.txt.

RFC 3265

Presence Optimizations for Push /Pull slides

CLAIMS

1. A method of providing information from an information delivery server (RLS) to a client (100) regarding a plurality of users (102), **characterised by** the following steps, executed in the information delivery server:
   - receiving a subscription request from the client for information on a set of users,
   - receiving information updates regarding the set of users, and storing updated user information in a database (106),
   - receiving a request for user information from the client, and
   - retrieving requested user information from the database and sending a notification to the client including the retrieved user information in response to the user information request.

2. A method according to claim 1, **characterised in** that the users are mobile users and the user information is presence information on the mobile users.

3. A method according to claim 1 or 2, **characterised in** that each user is connected to an associated host server (104) maintaining information on the user, and that said information updates regarding the set of users is received from associated host servers.

4. A method according to claim 3, **characterised in** that the information delivery server establishes a subscription for user information updates with each of the host servers associated with the set of users, in response to the received subscription request.

5. A method according to claim 4, **characterised in** that updated user information is pushed from the host servers to the information delivery server whenever the information on users has changed.

6. A method according to any of claims 3-5, **characterised in** that the information delivery server further acts as a host server maintaining information on the client.

7. A method according to any of claims 3-6, **characterised in** that the host servers further act as information delivery servers for providing user information to their associated users.

8. A method according to any of claims 1-7, **characterised in** that the requested user information is pulled from the information delivery server by the client.

9. A method according to any of claims 1-8, **characterised in** that the request for user information received from the client is limited to a subset of users comprising fewer users than the set of users.

10. A method according to claim 9, **characterised in** that the subset of users is adapted to a service and/or application currently utilised by the client.

11. A method according to any of claims 1-10, **characterised in** that the subscription request from the client includes a time of expiration.

12. A method according to any of claims 1-8, **characterised in** that the subscription request from the client includes a

minimum time ("throttle time") between successive notifications.

13. A method according to claims 11 and 12, **characterised in** that the minimum time between successive notifications corresponds to the time of expiration.

14. An information delivery server for providing information to a client regarding a plurality of users, **characterised by:**

- means for receiving a subscription request from the client for information on a set of users,
- means for receiving information updates regarding the set of users,
- a database for storing updated user information,
- means for receiving a request for user information from the client, and
- means for retrieving requested user information from the user database and for sending a notification to the client including the retrieved user information, in response to the user information request.

15. An information delivery server according to claim 14, **characterised by** means for establishing a subscription for user information updates, in response to a received subscription request, with one or more host servers maintaining information on associated users in the set of users.

16. An information delivery server according to claim 14 or 15, **characterised in** that the information delivery server is adapted to act as a host server maintaining information on the client.

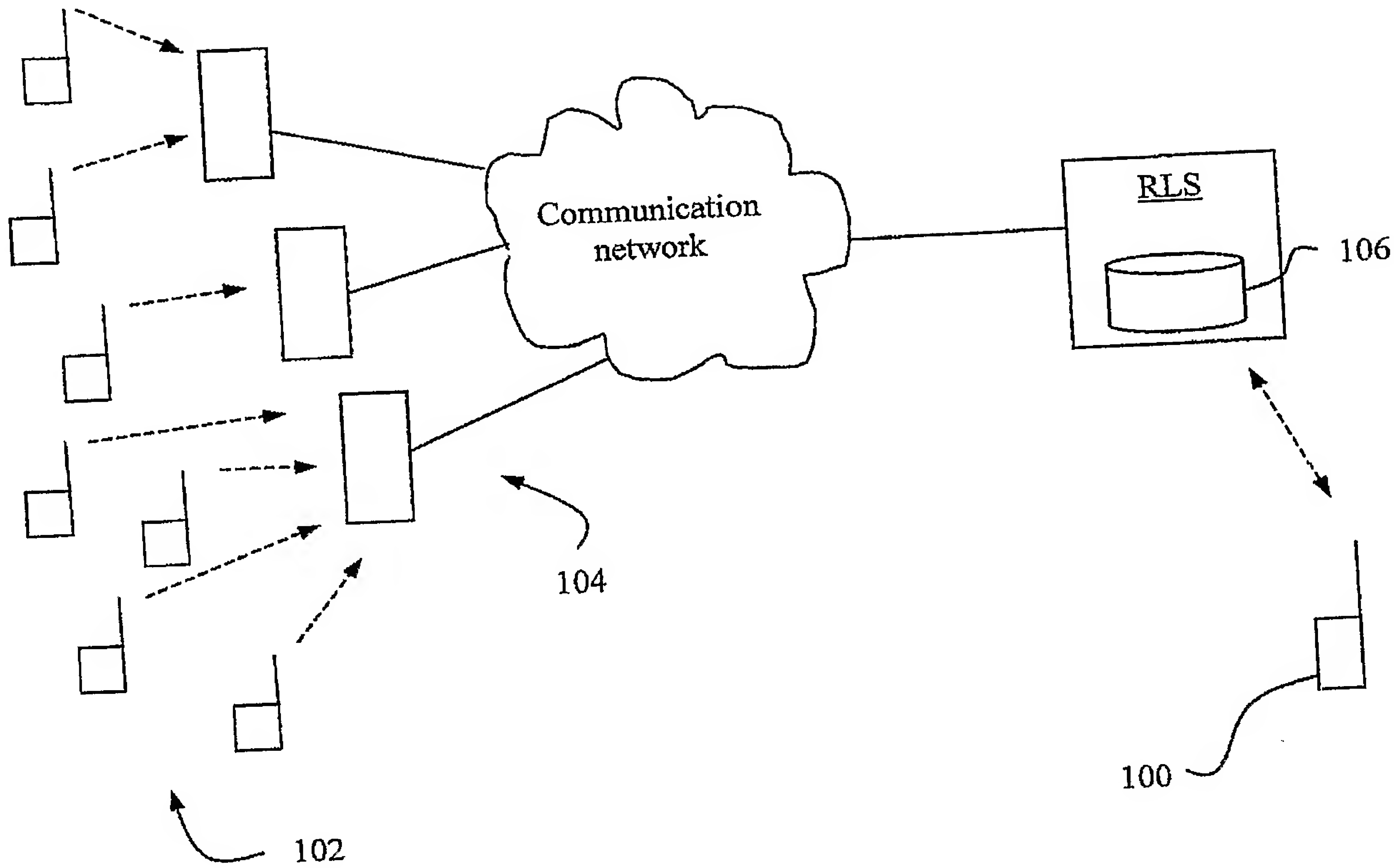

Figure 1